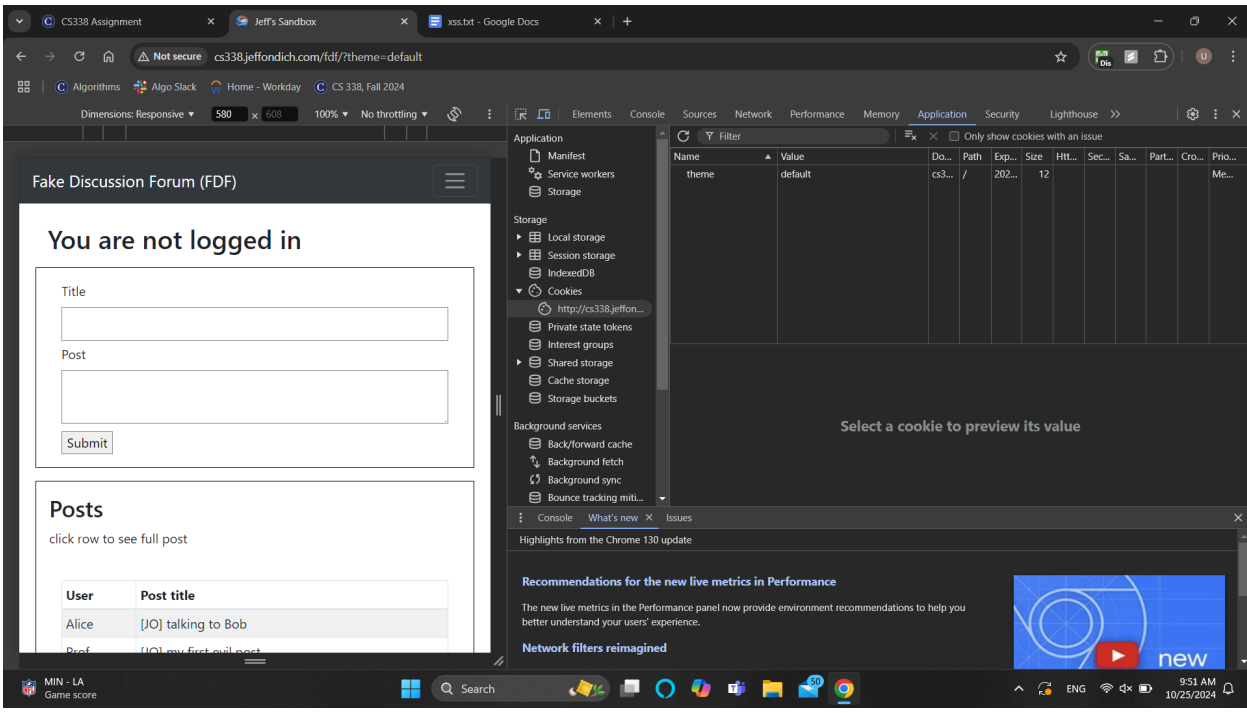
Ugo Anyaegbunam and Ntense Obono

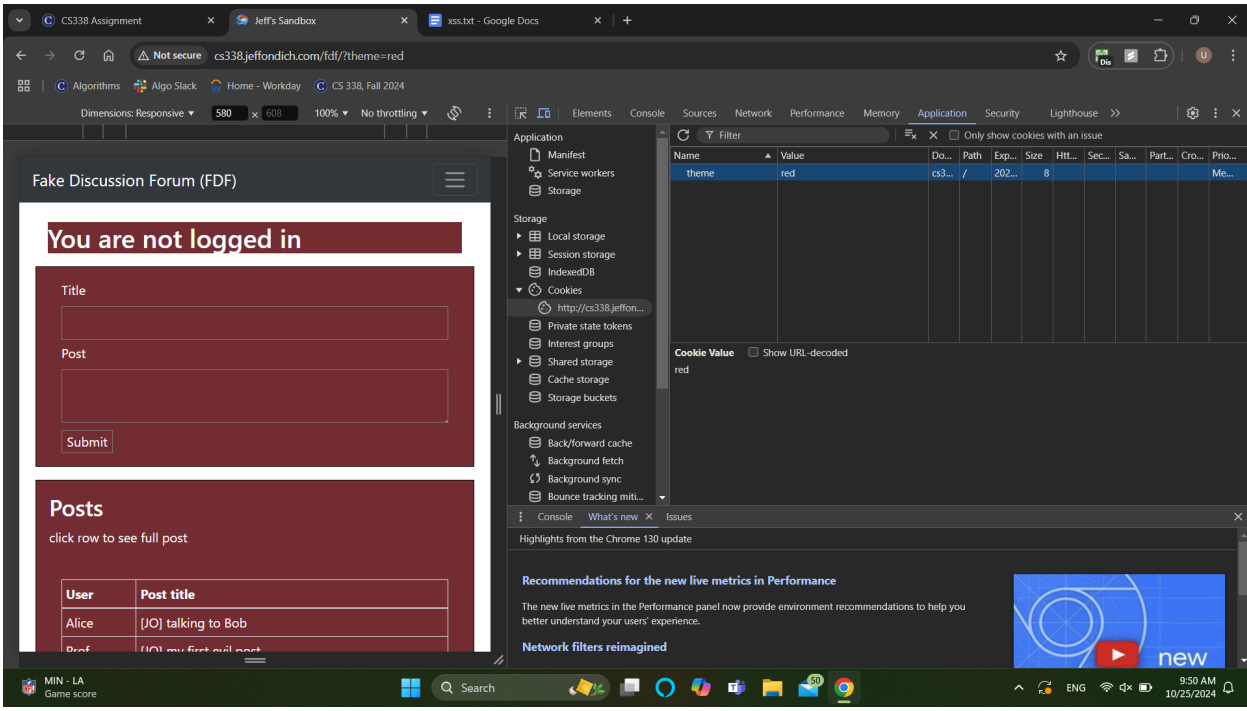
Part 1: Cookies

1. Go to FDF and use your browser's Inspector to take a look at your cookies for cs338.jeffondich.com. Are there cookies for that domain? What are their names and values?
    a. Yes, there are cookies. The only one I see is theme=default

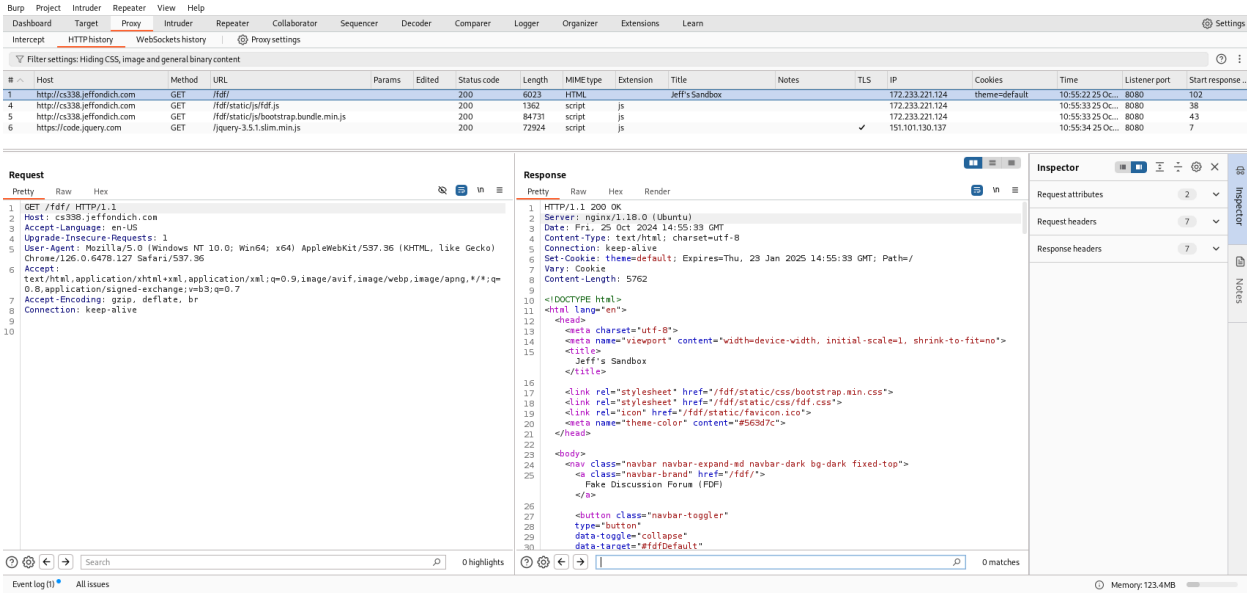


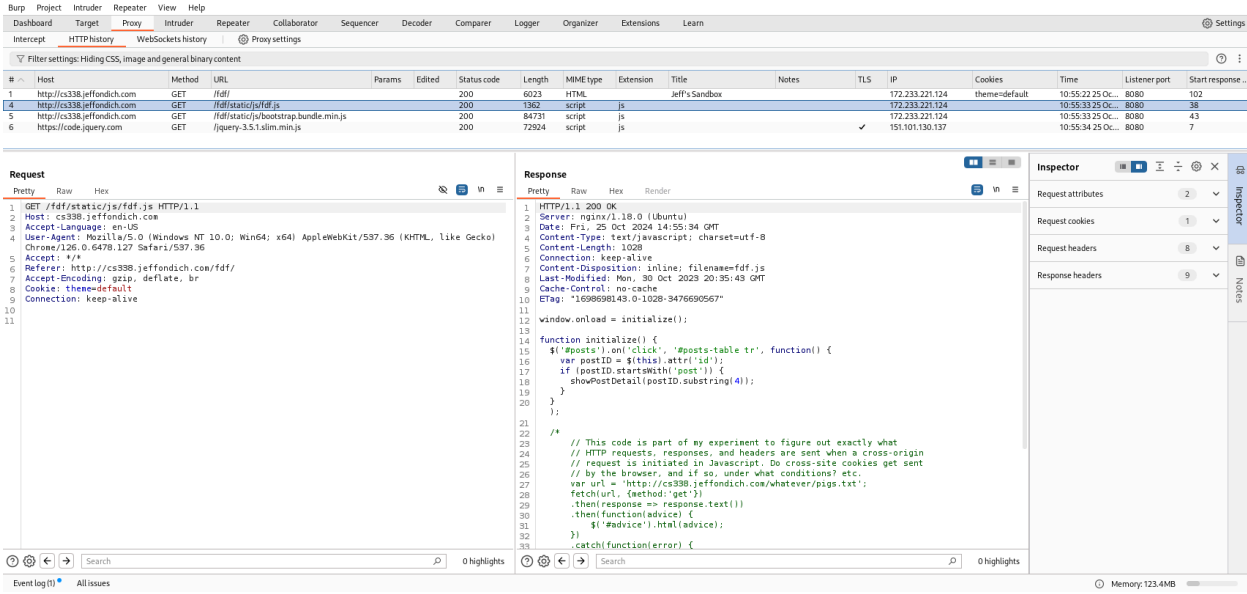
2. Using the "Theme" menu on the FDF page, change your theme to red or blue. Look at your cookies for cs338.jeffondich.com again. Did they change?

a. Yes

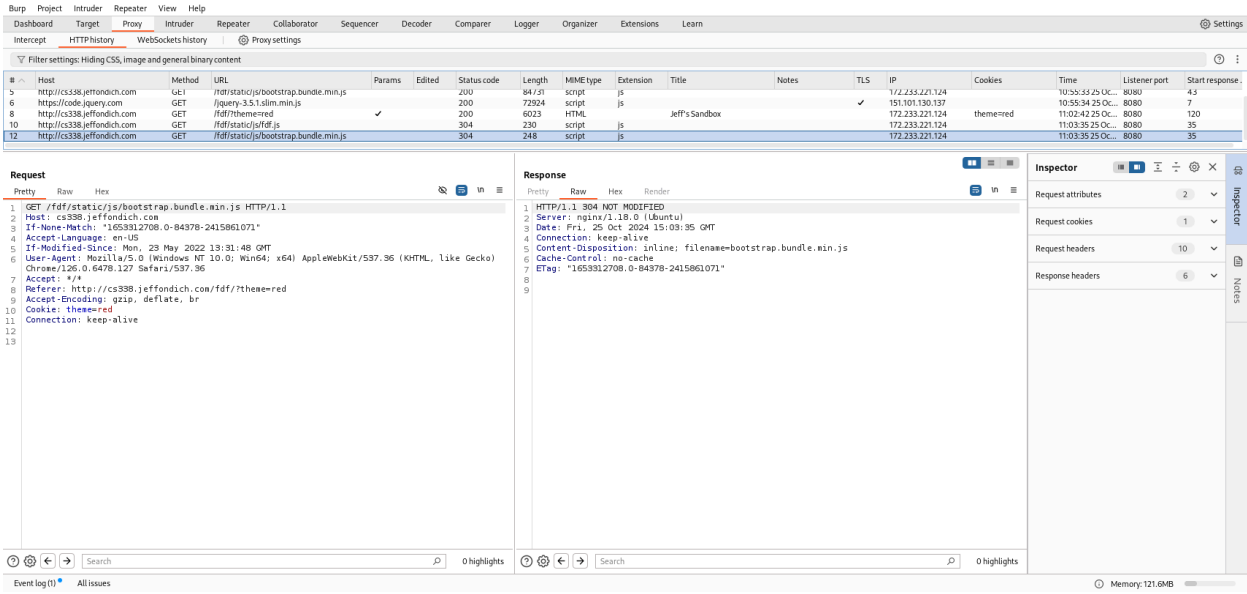


3. Do the previous two steps (examining cookies and changing the theme) using Burpsuite. What "Cookie:" and "Set-Cookie:" HTTP headers do you see? Do you see the same cookie values as you did with the Inspector?
    a. I see Set-Cookie: theme=default; Expires=Thu, 23 Jan 2025 14:55:33 GMT; Path=/ in the response for the get and then the next get contains Cookie: theme=default



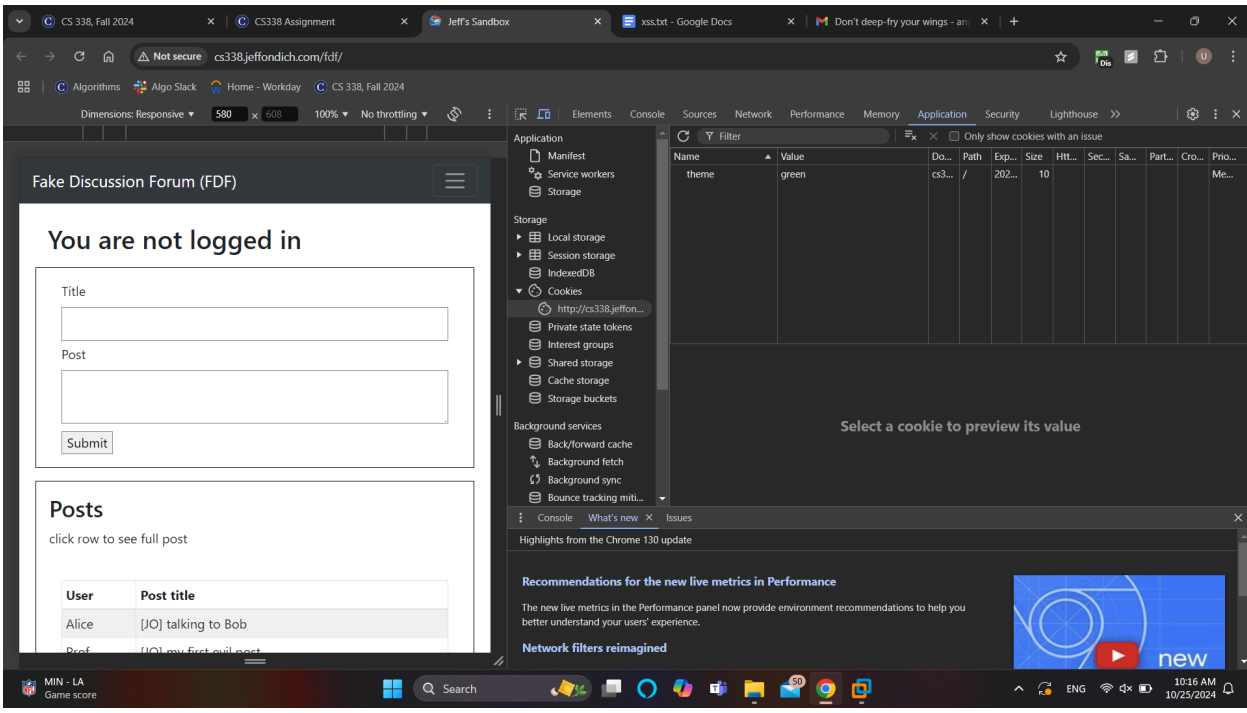


and then when I changed it the theme to red the get request looked like this:



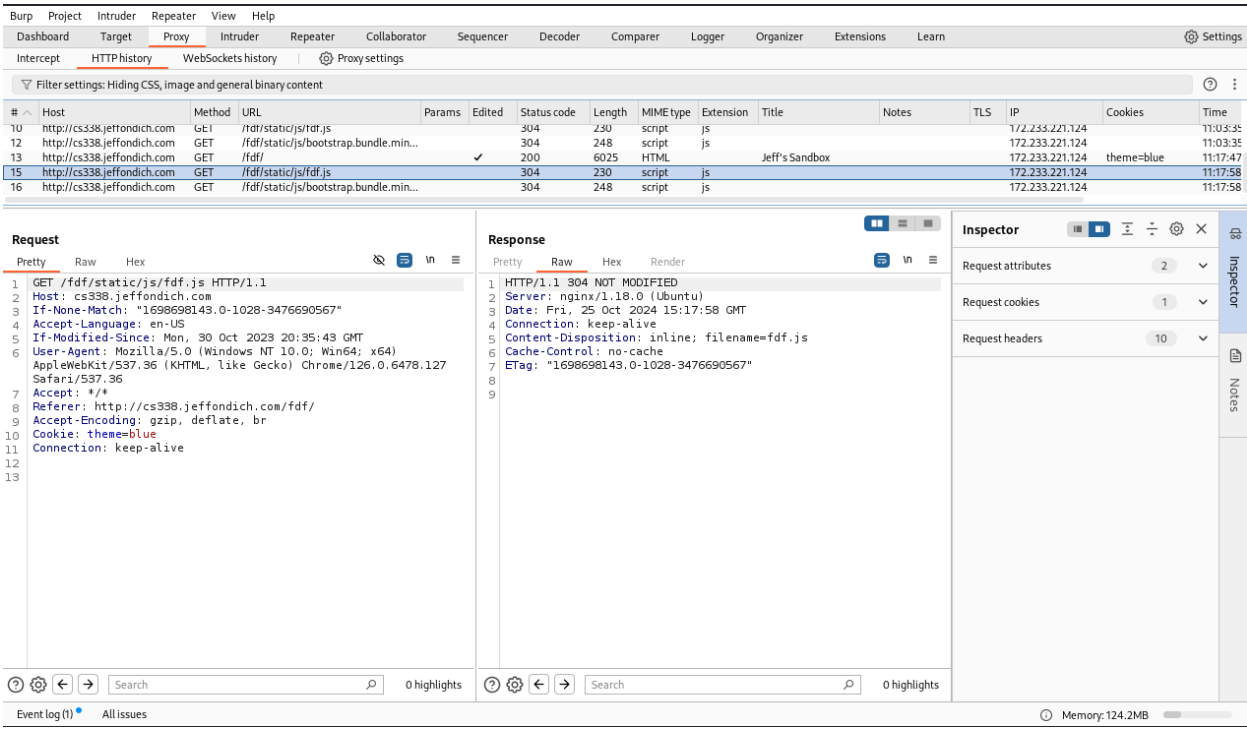
4. Quit your browser, relaunch it, and go back to the FDF. Is your red or blue theme (wherever you last left it) still selected?
    a. When I completely quit chrome it did not stay, but when i just closed the tab and came back to it, then it was there.
5. How is the current theme transmitted between the browser and the FDF server?
    a. In the GET request within the cookie header
6. When you change the theme, how is the change transmitted between the browser and the FDF server?

    a. In the GET request where we make the change, the response has Set-Cookie: theme=red; Expires=Thu, 23 Jan 2025 15:03:35 GMT; Path=/ and then every request after that has Cookie: theme=red
7. How could you use your browser's Inspector to change the FDF theme without using the FDF's Theme menu?
    a. You can go to where you were examining and actually just enter the value. I noticed that when you type a value that isn't in the menu, then it just goes to the default.



8. How could you use Burpsuite's Proxy tool to change the FDF theme without using the FDF's Theme menu?
    a. Intercept the request and change the theme value by hand in the Cookie header. Here you can see that the theme value contains blue without

seeing a GET /fdf/?theme=blue HTTP/1.1come before:



9. Where does your OS (the OS where you're running your browser and Burpsuite, that is) store cookies? (This will require some internet searching, most likely.)
    a. According to this website, They're located in the INetCookies folder in the C: drive on my WindowsOS.

Part 2: Cross-Site Scripting (XSS)

Steps to take:

- Login to the FDF as Alice (alice@example.com, password: alice) or Bob (bob@example.com, password: bob) or Eve (go ahead, guess her email and password!).
- Make a post and view your post by clicking on its title in the list of posts at the bottom of the page. Please include your initials in the title of your post like I did with my "[JO]" titles.
- Go back to the FDF home page.
- Click on each of Moriarty's posts and pay attention. What happens?
- Study the source code of each of Moriarty's posts. It's shown on the post details page itself, but you should also right-click on the background and select View Page Source to take a look at the raw HTML. Or, alternatively, you can select the Elements tab in the browser Inspector and take a look at the source. Regardless, your goal is to figure out how Moriarty made the FDF behave surprisingly.

- Experiment making your own posts as Alice, Bob, or Eve. Make the title descriptive of what you're trying to do, but fool around in the the post body however you want to. (If you're unfamiliar with HTML, CSS, and Javascript, you may want to grab a classmate who knows about those things to help you implement your nefarious plans.)

If last year's experience is any guide, somebody will mess up FDF unintentionally. If the site becomes unusable, Slack me and email me with details about the offending post, and I can go in and delete it.

Questions:

1. Provide a diagram and/or a step-by-step description of the nature and timing of Moriarty's attack on users of the FDF. Note that some of the relevant actions may happen long before other actions.
    a. Moriarty logs in, and makes a post containing malicious JavaScript. The post is now there and sitting in the database. Regardless of how much time has passed, when someone else like Alice or Bob login and view the post, the post is pulled from the database. When the post is pulled from the database, the bad JavaScript Moriarty put in there is evaluated and executed as well.
2. Describe an XSS attack that is more virulent than Moriarty's "turn something red" and "pop up a message" attacks. Think about what kinds of things the JavaScript might have access to via Alice's browser when Alice views the attacker's post.
    a. I don’t know the exact syntax for this, but what Moriarty could do is put a web query in his post. FDF doesn’t have the HTTP only tag for cookies, so what he could do is have code that makes a post request to his server containing data from where all the cookies are located, including session cookies. So then Moriarty would have people’s session cookies and wouldn’t even need their password to login if he intercepts the request with something like burp suite.
3. Do it again: describe a second attack that is more virulent than Moriarty's, but that's substantially different from your first idea.
    a. You could make it so that one of the buttons that users press all the time becomes an invisible hyperlink when your post is brought up. Then you can make it a link to a malicious webpage.
4. What techniques can the server or the browser use to prevent what Moriarty is doing?
    a. Parse inputs very carefully

b. Minimize access to files from the browser
c. Encrypt session cookies maybe?